Experiment Report: Hardware Comparison on Tiny Shakespeare (Character-Level nanoGPT)

Experimental Setup

Date: Tuesday, 02/03/2026
Participants: Youssif Salah, John Lee, Maximo Sanchez

In this experiment, we trained a **tiny character-level GPT model** on the *Tiny Shakespeare* dataset to evaluate whether differences in hardware specifications (CPU vs GPU, and across GPU tiers) lead to observable differences in training behavior at small model scales.

All participants trained the same model using the **nanoGPT framework** within the shared `COMP560-jmac` repository. To ensure experimental fairness and reproducibility, all runs were logged to a shared Weights & Biases project: **GPUvsCPUResearch** (https://wandb.ai/dickinson-comp560-sp26/GPUvsCPUResearch)

The experiment was conducted on three machines with differing hardware capabilities:

- **RTX 3070 GPU system**
- **RTX 4060 GPU system**
- **Apple M2 (CPU-only) system**

These systems were ordered conceptually from highest to lowest theoretical compute capability. All models were trained for the same number of iterations using identical configurations, seeds, and dataset splits.

Model Configuration (Held Constant Across All Runs)

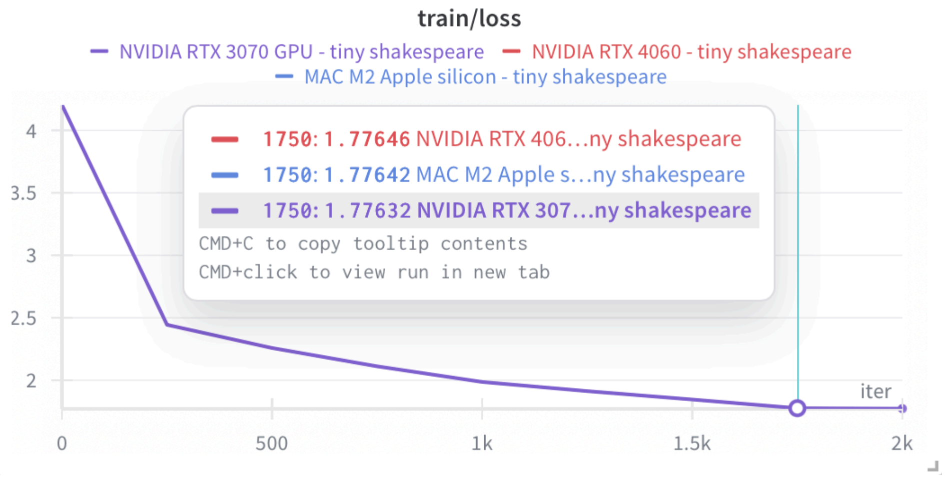
To isolate the effect of hardware, **all hyperparameters were fixed** across devices:

```
# embryonic GPT model
n_layer = 3
n_head = 3
n_embd = 126 # need n_embd % n_head == 0
dropout = 0.0
learning_rate = 1e-3 # with baby networks can afford to go a bit higher
max_iters = 2000
lr_decay_iters = 2000 # make equal to max_iters usually
min_lr = 1e-4 # learning_rate / 10 usually
beta2 = 0.99 # make a bit bigger because number of tokens per iter is small
warmup_iters = 100 # not super necessary potentially
```

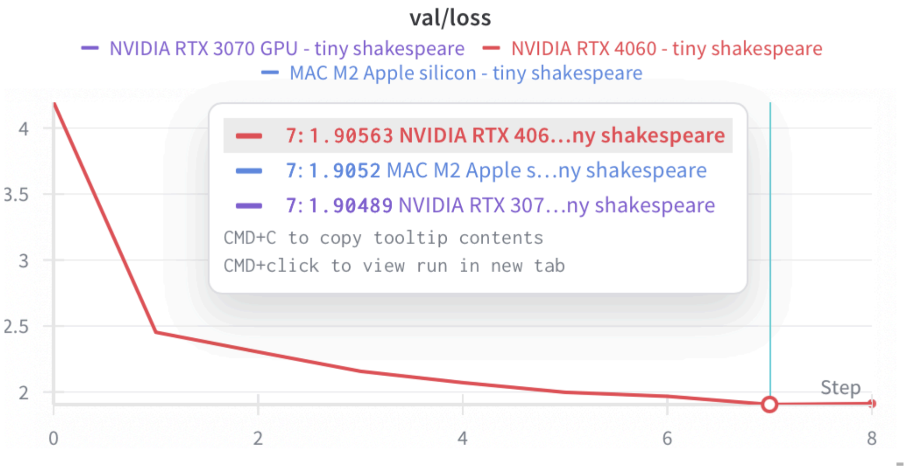
The model is intentionally shallow and small, representing a **low-capacity transformer** designed to overfit quickly on a character-level dataset.

Results and Observations

Across all three machines, **training loss and validation loss curves were nearly identical**. Minor fluctuations were observed between runs as seen in figures 1 & 2; however, these variations were **not correlated with hardware capability**.



▲ Figure 1



▲ Figure 2

Notably:

- The RTX 4060 occasionally achieved slightly lower loss than the RTX 3070.
- The Apple M2 CPU-only run performed comparably to both GPU-based systems.
- Differences in loss values were minimal and well within expected stochastic noise.

At this scale, the model converges smoothly and predictably, and hardware advantages in parallelism did not translate into improved optimization outcomes.

These results indicate that for small transformer models with low depth and moderate learning rates, the optimization landscape is stable enough that different hardware platforms converge to essentially the same solution.

Interpretation

The lack of observable divergence is expected and highlights an important insight:

> **Hardware affects training speed, not training dynamics, when the optimization problem is simple and numerically stable.**

At shallow depth:

- Gradients are well-conditioned
- Floating-point error does not accumulate significantly
- Attention mechanisms are simple and robust
- Optimization follows a deterministic trajectory

As a result, even significantly different hardware platforms produce nearly identical loss curves.

This demonstrates that **high-end hardware is not necessary** for training or experimenting with small character-level language models.

Conclusion

This experiment shows that when training a low-depth, low-capacity transformer, differences in hardware specifications (GPU tier vs CPU-only systems) do **not meaningfully impact model convergence or final loss values**. At this scale, the optimization problem is sufficiently stable that all systems reach similar solutions, regardless of computational power.

These findings reinforce the idea that **hardware advantages become relevant only when numerical instability, scale, or optimization difficulty are introduced**. For introductory models and small-scale experiments, accessible hardware is sufficient for achieving equivalent learning outcomes.

Next Steps: Inducing Hardware-Sensitive Divergence

To move beyond this stable regime, our next experiments will intentionally **amplify sources of instability** to observe when and how hardware differences begin to matter.

Planned Experiment 2: Increased Model Depth

```
n_layer = 8
n_head = 6
n_embd = 192
```

Rationale:
Deeper transformers:

- Accumulate floating-point error across layers
- Are more sensitive to attention drift
- Amplify numerical differences between fp32 (CPU) and mixed precision (GPU)

Expected Outcome:
Loss curves between CPU and GPU runs will visibly diverge, even with identical seeds and configurations.

Planned Experiment 3: Aggressive Learning Rate

```
learning_rate = 3e-3
warmup_iters = 0
```

Rationale:
A larger learning rate introduces chaotic optimization dynamics, where:

- Small numeric differences influence gradient direction
- Different hardware may fall into different local minima
- Mixed-precision GPU runs diverge faster than fp32 CPU runs

Expected Outcome:
Some runs may converge smoothly while others diverge early, despite identical configurations — a phenomenon commonly observed in large-scale training.

Final Note

These next experiments aim to transition from a **hardware-invariant regime** to a **hardware-sensitive regime**, allowing us to directly study how numerical precision, depth, and optimization stability interact with computer architecture.